# WP10-H — Sovereign Sync Protocol

Appendix H of the Sovereign Memory Initiative

Author: Neofirebird (Brad Donwen)

Version: 1.0

License: CC BY 4.0

File: `WP10-H\_Sovereign\_Sync\_Protocol.md`

---

## 1. Abstract

Sovereign Sync is a manual memory workflow that enables users to simulate persistent memory, version control, and cross-session continuity with large language models—without plugins, API hooks, or automated agents. Designed for constrained environments like ChatGPT, it equips users with a disciplined method to externalize, compress, rehydrate, and audit project states using only file management and prompt design.

At its core, Sovereign Sync formalizes a loop: capture state; compress to symbolic frames; store with explicit naming; manually rehydrate at session start; and progressively reconcile outputs into durable artifacts. This appendix (WP10-H) operationalizes the method and extends it across single- and multi-agent contexts, including a symbolic mode add-on, with practical templates for immediate adoption.

---

## 2. Problem Definition

Modern LLMs provide remarkable in-session memory but struggle with continuity across sessions and context windows. Users face:

1) Memory caps and obsolescence

- Context windows are finite; older content is forgotten or truncated.

- Session loss resets state; unstated assumptions are not recoverable.

- Model variants and updates change behavior, introducing drift.

2) Loss of continuity

- No canonical source of truth persists across chats.

- Goals and decisions diffuse across threads with no lineage.

- Iteration lacks robust versioning, making regression recovery difficult.

3) Over-reliance on automation

- Users assume plugins, agent modes, or vector databases are prerequisites for persistence.

- When such tools are unavailable, projects stall or bloat with repeated re-priming.

Sovereign Sync addresses these constraints with a manual, user-sovereign protocol for continuity, compression, and reconstruction.

---

## 3. Manual Solution (Sovereign Sync Logic)

Sovereign Sync is a deterministic, human-in-the-loop state machine. It transforms volatile conversational state into external, stable artifacts and provides a repeatable method to re-inject these artifacts to restore context.

Core logic:

- Externalize: Export conversation deltas into structured artifacts (`.md` modules, `.json` manifests).

- Compress: Use symbolic scaffolds to reduce high-entropy prose into low-entropy frames.

- Version: Attach explicit, sortable, and human-readable IDs.

- Rehydrate: Bootstrap new sessions by feeding compact state packets via manual prompts.

- Reconcile: Merge new outputs into the source of truth with changelogs and semantic diffs.

- Budget: Respect context limits via slot allocation and frame granularity.

Sovereign Sync is intentionally non-automated: a disciplined ritual that keeps the user in control of memory formation and identity across agents and time.

---

## 4. Core Components

### 4.1 Naming Conventions

A robust naming scheme prevents ambiguity and accelerates manual retrieval. Use sortable, semantic, and stable identifiers.

Pattern:

`<PROJECT>::<STREAM>::<ARTIFACT>::<VERSION>::<STAMP>.<EXT>`

Field guidelines:

- `PROJECT`: Short, stable project code, e.g., `WP10`.

- `STREAM`: Functional lane, e.g., `ARCH`, `RESEARCH`, `DRAFT`, `OPS`.

- `ARTIFACT`: Type or role, e.g., `STATE`, `BRIEF`, `PROMPT`, `PLAN`, `GLOSSARY`.

- `VERSION`: Major.Minor.Patch with zero padding, e.g., `v01.02.00`.

- `STAMP`: ISO datetime or epoch, or increment counter, e.g., `2025-01-15T21-40Z`.

- `EXT`: `md`, `json`, `txt`, `yaml`, `csv`.

Examples:

- `WP10::ARCH::STATE::v01.00.00::2025-01-15T21-40Z.md`

- `WP10::DRAFT::OUTLINE::v02.01.00::2025-01-15T22-05Z.md`

- `WP10::OPS::CHANGELOG::v00.00.07::2025-01-15T22-10Z.md`

- `WP10::ARCH::MANIFEST::v01.00.00::2025-01-15T22-10Z.json`

Call-out: Prefer few, durable streams. Avoid renaming files; increment versions.

### 4.2 Upload/Download Loop

The Sovereign Sync loop has five steps:

1) Capture

- After a meaningful session, copy key outputs into a `STATE` or `MANIFEST` artifact.

- Extract decisions, constraints, open questions, and artifacts.

- Keep raw transcripts minimal; focus on distilled state.

2) Compress

- Apply symbolic frames (see Appendix H-2) to summarize goals, roles, assets, and deltas.

- Maintain both a human-readable `.md` state and a machine-oriented `.json` manifest.

3) Version

- Save as a new version. Update `CHANGELOG` with concise semantic diffs.

- Never overwrite; always append version and stamp.

4) Rehydrate

- At the start of a new session, paste the `Rehydration Prompt` plus the latest `STATE` excerpt.

- If the state is large, load in prioritized chunks according to budget.

5) Reconcile

- After the session, update the state artifacts with merged changes and new decisions.

- Note trade-offs, conflicts, and accepted re-prioritizations.

Diagram (textual):

[User] -> (Chat Session) -> [Outputs]

[Outputs] -> (Distill + Compress) -> [`STATE.md`, `MANIFEST.json`]

[`STATE.md`, `MANIFEST.json`] -> (Version) -> [Repository]

[Repository] -> (Rehydrate via prompt) -> (New Chat Session)

### 4.3 Manual Rehydration Prompts

A rehydration prompt is a compact scaffold that instructs the model to adopt the project state.

Template:

```

System: You are assisting with <PROJECT> using the Sovereign Sync method.

User: Rehydrate from the following state packets. Acknowledge with a one-paragraph summary and list any missing dependencies.

<STATE::summary\_block>

<STATE::goals\_block>

<STATE::constraints\_block>

<STATE::assets\_block>

<STATE::open\_questions\_block>

<MANIFEST::index\_block> # optional

```

For very tight budgets, use a micro-primer:

```

User: Sovereign Sync rehydrate (micro).

Context:

- Project: <PROJECT>

- Goals (3): <...>

- Decisions (3): <...>

- Constraints (3): <...>

- Active artifacts: <IDs>

- Next actions: <3 bullets>

Respond with: summary (<=80 words), risks (<=3 bullets), and clarifying questions (<=3).

```

Include a brief of the naming scheme so the model can request artifacts by ID.

### 4.4 Memory Slot Budgeting

Given a finite context window, allocate “slots” to state frames. Think in tokens, not pages.

Recommended slot map (adjust per model):

- 10% Protocol primer (Sovereign Sync instructions)

- 30% Project state core (goals, decisions, constraints)

- 20% Key artifacts (outline, schema, glossary)

- 20% Current task context (prompt, examples)

- 10% Safety buffer (clarifications, iteration)

- 10% Slack for emergent details

Compression heuristics:

- Convert prose into bullet frames.

- Use IDs and references instead of re-pasting full artifacts.

- Keep a short `GLOSSARY` to reduce repeated definitions.

- Evict low-variance details; store them externally; re-load on demand.

---

## 5. Single-Agent Application

Scenario: A single conversational thread with one model variant (e.g., ChatGPT-4/4o).

Workflow:

1) Initialize

- Create `PROJECT::ARCH::MANIFEST.json` and `PROJECT::ARCH::STATE.md`.

- Seed `GLOSSARY`, `CONSTRAINTS`, `GOALS`, and `DECISIONS`.

2) Work Session

- Start with a rehydration prompt and relevant state slices.

- Perform a focused task (e.g., outline chapter, design schema).

- Return tangible artifacts: `OUTLINE`, `SPEC`, `DRAFT`.

3) Distill

- Extract decisions, new constraints, and updated goals.

- Update `STATE.md` and `MANIFEST.json` with the delta.

- Log semantic diff in `CHANGELOG.md`.

4) Version and Commit

- Save new versioned files.

- Optionally tag milestones, e.g., `MVP`, `ALPHA`, `H1`.

5) Periodic Compaction

- If `STATE.md` grows, archive older sections with references.

- Maintain a rolling `STATE\_MIN.md` for quick rehydration.

Quality gates:

- No task accepted without accompanying delta in `STATE` and `CHANGELOG`.

- Each artifact has an owner note and purpose line.

- Questions unresolved are tracked in `OPEN\_QUESTIONS`.

---

## 6. Multi-Agent Expansion (Appendix H-1)

When using multiple models or parallel threads, Sovereign Sync extends into a cross-agent protocol that prevents divergence.

Principles:

- One source of truth: `ARCH::STATE.md` and `ARCH::MANIFEST.json`.

- Many views: agent-specific `PROMPT` frames and task-scoped `STATE\_VIEW` extracts.

- Explicit handoffs: standardized `HANDOFF.md` packets between agents.

Components:

- `AGENT\_PROFILE.md`: Capabilities, preferred input format, output style, known pitfalls.

- `TASK\_BRIEF.md`: Objective, constraints, acceptance criteria, timebox.

- `STATE\_VIEW.md`: Minimal slice of `STATE` relevant to the task.

- `HANDOFF.md`: What was done, evidence, deltas proposed, questions back.

Handoff template:

```

HANDOFF

- Task ID: <PROJECT>::<STREAM>::<TASK>::<v>

- Inputs: <STATE\_VIEW IDs>, <Artifacts>

- Process: <Method used / prompts summary>

- Outputs: <Artifacts IDs>

- Proposed Deltas: <Decisions, Constraints, Glossary entries>

- Open Questions: <...>

- Risks/Assumptions: <...>

```

Merge protocol:

1) Agent B receives `HANDOFF` + `STATE\_VIEW`.

2) Performs task; proposes deltas.

3) User reviews; merges accepted deltas into `STATE` and `MANIFEST`.

4) Record merge in `CHANGELOG` with attributions.

Conflict resolution:

- Prefer explicit decisions over implicit patterns.

- If two agents conflict, create `DECISION\_RECORD` with rationale and trade-offs.

- Update `CONSTRAINTS` and `GLOSSARY` to lock in the resolution.

---

## 7. Symbolic Mode Add-on (Appendix H-2)

Symbolic Mode compresses conversational entropy into stable, low-token symbols. It is optional but powerful for budgeted contexts and consistent rehydration.

Symbols:

- R: Role frames (e.g., `R/Editor`, `R/Architect`)

- G: Goal frames (`G/Primary`, `G/Secondary`)

- C: Constraint frames (`C/Security`, `C/Scope`)

- D: Decision frames (`D/Accepted`, `D/Deferred`)

- A: Asset frames (`A/Outline/v02`, `A/Schema/v03`)

- Q: Question frames (`Q/Priority`, `Q/Blocked`)

- P: Process frames (`P/Method`, `P/Checklists`)

- X: Risk frames (`X/Assumption`, `X/Dependency`)

Encoding examples:

- `R/Architect: Owns information model; defines schemas; approves deltas.`

- `G/Primary#1: Deliver Appendix H with templates.`

- `C/Scope#2: No automation; manual prompts only.`

- `D/Accepted#5: Adopt versioned naming pattern.`

- `A/Outline#v02: Sections 1–10 approved; pending Appendices.`

- `Q/Priority#3: Open: token budget policy for multi-agent handoffs.`

- `P/Checklist#Rehydrate: Confirm goals, deltas, artifacts, next actions.`

- `X/Dependency#LLM: Model window 32k; prefer symbolic compression.`

Rehydration in Symbolic Mode:

```

User: Sovereign Sync | Symbolic Mode

Load frames:

R: Architect, Editor

G: Primary#1, Secondary#1-2

C: Scope#2, Budget#1

D: Accepted#5, Deferred#2

A: Outline#v02, Glossary#v01

Q: Priority#3

Respond:

- 60-word situational summary

- Missing frames (IDs)

- Task plan (<=5 bullets)

```

Benefits:

- Predictable token footprint

- Easier diffs and merges

- Clear reference semantics for agent coordination

---

## 8. Future Considerations (Integration with Agent Mode, GPT-5)

Sovereign Sync is designed to interoperate with—but not depend on—automation.

Paths forward:

- Agent Mode bridge: Agents can ingest `MANIFEST.json` and `STATE\_VIEW.md` to operate within a user-sovereign context, emitting structured `HANDOFF` and `DELTA` packets.

- Memory APIs: When models expose persistent memory, map Sovereign Sync frames to memory slots, keeping `STATE` as canonical.

- GPT-5 and larger windows: Expand slot budgets but keep symbolic discipline to maintain determinism.

- Tool use: Optional semantic diff tools can assist merges; the protocol remains functional with manual inspection.

- Provenance: Add `SIGNOFF.md` for approvals, attributions, and audit trails.

Non-goals:

- Replacing human judgment

- Opaque agent-to-agent negotiation without user oversight

---

## 9. Conclusion + Use Cases

Sovereign Sync restores continuity, precision, and control in constrained LLM environments. By externalizing state, compressing symbolically, and rehydrating via disciplined prompts, users achieve persistent memory effects without automation.

Representative use cases:

- Long-form writing: Maintain outlines, style guides, and chapter decisions across months.

- Research programs: Track hypotheses, evidence, and evolving claims with reproducible deltas.

- Product design: Coordinate specs across design, engineering, and QA agents.

- Policy drafting: Ensure decisions and constraints remain stable across revisions.

- Coaching and learning: Persist goals, curricula, and progress milestones.

Sovereign Sync is a philosophy of sovereignty—humans own the memory, models serve the process.

---

## 10. Appendices + Templates

### A. Directory Layout

```

/SovereignSync/

/PROJECT\_WP10/

/ARCH/

WP10::ARCH::STATE::v01.00.00::<STAMP>.md

WP10::ARCH::STATE\_MIN::v01.00.00::<STAMP>.md

WP10::ARCH::MANIFEST::v01.00.00::<STAMP>.json

WP10::ARCH::GLOSSARY::v01.00.00::<STAMP>.md

WP10::OPS::CHANGELOG::v00.00.01::<STAMP>.md

/DRAFT/

WP10::DRAFT::OUTLINE::v01.00.00::<STAMP>.md

WP10::DRAFT::SECTION-1::v00.01.00::<STAMP>.md

/AGENTS/

AGENT\_PROFILE\_\_Editor.md

AGENT\_PROFILE\_\_Architect.md

/TASKS/

TASK\_BRIEF\_\_Outline\_v02.md

STATE\_VIEW\_\_OutlineCore\_v02.md

HANDOFF\_\_Outline\_to\_Editor\_v02.md

```

### B. MANIFEST.json (example)

```json

{

"project": "WP10",

"version": "v01.00.00",

"artifacts": {

"STATE": "WP10::ARCH::STATE::v01.00.00::2025-01-15T22-10Z.md",

"OUTLINE": "WP10::DRAFT::OUTLINE::v02.00.00::2025-01-15T22-12Z.md",

"GLOSSARY": "WP10::ARCH::GLOSSARY::v01.00.00::2025-01-15T22-11Z.md"

},

"constraints": [

"Manual-only; no plugins or APIs",

"Symbolic compression preferred",

"All changes logged in CHANGELOG"

],

"open\_questions": [

"What is the minimal viable STATE\_MIN token footprint?",

"Preferred conflict resolution mechanism for agent deltas?"

],

"agents": [

{"name": "Architect", "profile": "AGENT\_PROFILE\_\_Architect.md"},

{"name": "Editor", "profile": "AGENT\_PROFILE\_\_Editor.md"}

]

}

```

### C. STATE.md (skeleton)

```

# STATE — <PROJECT>

Version: v01.00.00 | Stamp: <ISO>

Owner: <Name> | Method: Sovereign Sync

## Goals

- G1: ...

- G2: ...

## Constraints

- C1: Manual-only

- C2: Budget: <token/window>

- C3: ...

## Decisions

- D1: ...

- D2: ...

## Assets

- A1: <artifact ID> — purpose

- A2: ...

## Open Questions

- Q1: ...

- Q2: ...

## Current Focus

- F1: <task objective>

- Next actions: <3–5 bullets>

## Notes

- N1: ...

```

### D. CHANGELOG.md (skeleton)

```

# CHANGELOG — <PROJECT>

## v01.00.00 — <STAMP>

Added: Initial STATE, MANIFEST, GLOSSARY

Decisions: D1–D3

Constraints: C1–C2

Notes: Established Sovereign Sync workflow

## v01.01.00 — <STAMP>

Changed: Outline v02

Added: Glossary entries GLOS-004–006

Removed: Deprecated draft sections 0.1

```

### E. Rehydration Prompt (standard)

```

System: You are assisting with <PROJECT> under the Sovereign Sync protocol.

User: Rehydrate from state packets below. Then:

1) Return a 100-word situational summary.

2) List missing or stale elements.

3) Propose a task plan (<=6 bullets) aligned to current goals.

-- STATE\_MIN (paste) --

-- MANIFEST index (optional) --

-- TASK\_BRIEF (optional) --

```

### F. Rehydration Prompt (symbolic)

```

User: Sovereign Sync | Symbolic Mode

Frames loaded: R[], G[], C[], D[], A[], Q[]

Respond with summary (<=60 words), risks (<=3), questions (<=3), plan (<=5).

```

### G. Task Brief Template

```

TASK\_BRIEF — <PROJECT>::<STREAM>::<TASK>::<v>

Objective: ...

Scope: ...

Acceptance Criteria: ...

Constraints: ...

Inputs: <STATE\_VIEW IDs, Artifacts>

Deliverables: <Artifacts, Formats>

Timebox: <duration>

Notes: ...

```

### H. STATE\_VIEW Template

```

STATE\_VIEW — <ID>

Purpose: Provide minimal context for <task>.

Includes: Goals (subset), Constraints (subset), Decisions (subset), Assets (IDs).

Excludes: Non-relevant history.

```

### I. Manual Diff Checklist

- Identify decisions made this session.

- Identify constraints added/modified.

- Capture new assets and their IDs.

- Update glossary if terminology shifted.

- Record unresolved questions.

- Increment versions and stamps.

- Update `CHANGELOG` with semantic highlights.

---

Call-out: Sovereign Sync is intentionally lightweight. If you lose all but `STATE\_MIN.md` and one `OUTLINE`, you can still recover momentum in the next session. That is sovereignty.

End of document.